Fast Sinkhorn Filters: Using Matrix Scaling for Non-Rigid Shape Correspondence with Functional Maps

Gautam Pai[1] Jing Ren[2] Simone Melzi[3] Peter Wonka[2] Maks Ovsjanikov[1]
[1]LIX, Ecole Polytechnique, IP Paris [2]KAUST [3]Sapienza University of Rome

Abstract

In this paper, we provide a theoretical foundation for pointwise map recovery from functional maps and highlight its relation to a range of shape correspondence methods based on spectral alignment. With this analysis in hand, we develop a novel spectral registration technique: Fast Sinkhorn Filters, which allows for the recovery of accurate and bijective pointwise correspondences with a superior time and memory complexity in comparison to existing approaches. Our method combines the simple and concise representation of correspondence using functional maps with the matrix scaling schemes from computational optimal transport. By exploiting the sparse structure of the kernel matrices involved in the transport map computation, we provide an efficient trade-off between acceptable accuracy and complexity for the problem of dense shape correspondence, while promoting bijectivity.[1]

1. Introduction

Non-rigid shape matching remains at the core of many computer vision tasks including statistical shape analysis [4], texture mapping [10], and deformation transfer, [41], among others.

Among many existing approaches for this problem [43, 38], a prominent overall strategy is to exploit spectral quantities, such as the eigenfunctions of the Laplace-Beltrami operator, which are naturally invariant to isometric shape deformations. Within this category, the functional map framework, introduced in [28] proposes an efficient way to represent and compute mappings and achieves the state-of-the-art accuracy in difficult shape matching problems [23].

One of the advantages of this framework is that it allows to formulate shape correspondence as a simple optimization problem relating the basis functions on the two shapes, from which a dense point-to-point correspondence can be extracted. This framework has been successfully applied in both axiomatic [28, 16, 14, 32] and learning-based [18, 35, 13, 8] settings. However, one of the recurring issues of virtually all works in this domain, is defining and using the exact relation between functional and pointwise maps. This problem is especially relevant in the *conversion* step from functional to point-to-point correspondences. This conversion has been specifically treated in several works, including [28, 33, 11, 34, 15, 45, 23] among others. Despite this significant effort, the precise rigorous relation between the two representations still remains ill-defined. In this paper, we provide a rigorous theoretical justification for pointwise conversion from a functional map, and discuss how it is related to the problem of aligning the spectral embeddings of non-rigid shapes. We highlight that unlike a functional map, which is not well-suited for pointwise conversion, the *adjoint* operator can naturally be used for point-to-point map extraction both theoretically in the smooth setting, and in practice on discrete shapes.

With this foundation in hand, we propose a general framework for iterative spectral alignment, and illustrate that many previous shape matching methods are regularized variants of our meta algorithm. Finally, we introduce an effective regularized procedure using Sinkhorn's algorithm to compute accurate near-bijective pointwise correspondences that can scale to densely-sampled shapes. We find that existing approaches do not allow the recovery of an accurate, smooth and bijective point-to-point correspondence with acceptable time and memory complexity. Such methods are either too inaccurate (e.g., nearest-neighbor),

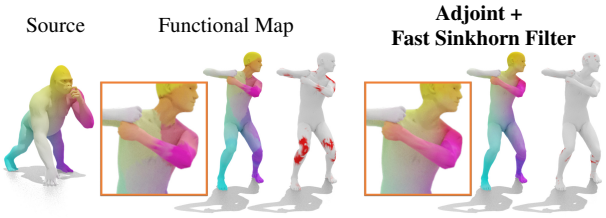

Figure 1: We formally justify pointwise map recovery from functional maps using the adjoint, and introduce *Fast Sinkhorn Filters:* an efficient method promoting bijectivity in this process. This yields better pointwise maps (color transfer) with improvement in bijectivity (errors in red).

[1]Demo code: https://github.com/paigautam/CVPR21_FastSinkhornFilters

or too time and memory consuming (e.g., linear-assignment solvers) thereby making them infeasible for practical applications.

We use our analysis of spectral alignment to construct a sparse kernel assignment matrix which is then efficiently processed using matrix scaling to output an entropic regularized transport plan. Finally by extracting the maximum likelihood estimate of this plan, we demonstrate that our approach, termed Fast Sinkorn Filters, produces accurate results often at a fraction of the cost of existing methods in both direct and iterative pointwise conversion applications.

2. Related Work

Shape matching is a very well-studied area of computer vision and computer graphics and its full overview is beyond the scope of our paper. Below we review the work most closely related to ours and focus primarily on the functional maps framework. We refer the interested readers to recent surveys including [43, 42, 3, 38] for an in-depth treatment of other shape matching approaches.

Functional Maps Our work focuses primarily on the functional map framework, which was introduced in [28] for solving near-isometric shape correspondence problems, and extended in many follow-up works, including [16, 1, 17, 32, 11, 6] among others (see also [29] for a general overview). The key idea in these techniques is to estimate linear transformations between spaces of real-valued functions, represented in a reduced functional basis. This linear structure implies that functional maps can be conveniently encoded as small matrices and optimized for using standard linear algebraic techniques.

In addition to the convenience of the representation itself, it has been observed by several works in this domain that many natural properties on the underlying pointwise correspondences can be expressed as objectives on functional maps [16, 36, 32, 6]. For example, orthonormal functional map matrices correspond to locally volume preserving maps [28, 16, 36], near isometries must result in functional maps that commute with the Laplacian [28, 46, 32, 20, 19], while conformal maps must preserve certain functional inner products [36, 6, 47].

These results typically *assume* that the functional map is induced as the pull-back of some underlying point-to-point correspondence. However, the space of linear functional transformations is strictly larger, which means that additional regularization is required. In [27, 26] the relation between pointwise and functional maps was studied and the authors proposed an optimization term [27] aimed to promote *only* functional maps arising from point-to-point ones. That work still used the default conversion scheme from [28], however.

Functional Map Conversion More closely related to ours are works that directly consider the question of point-to-point correspondence recovery from functional maps. This step is instrumental in all functional maps-based correspondence methods. As we highlight below, the original method [28] suggested a recovery technique based on considering images of indicator functions at points and then an efficient method based on iterative closest point in the spectral domain. Unfortunately, no justification or analysis was provided for whether this procedure has any analogue in the smooth setting.

Several follow-up works noted that the conversion step can have a fundamental limiting effect on the accuracy of the recovered maps [33, 11, 34, 15, 45, 23]. This has led to algorithms that incorporate smoothness using Coherent Point Drift in the spectral domain [33], penalizing spurious high-frequencies during conversion [10] and using higher-order objectives such as maximizing kernel density [44, 45], among others. Nevertheless, despite this significant effort the fundamental question of the relation between functional and pointwise maps in the smooth setting (i.e., independently of the shape discretization) remains open. This is unfortunate, as for example, discrete differential geometry operators [24] such as the Laplacian are discretized precisely using principles from the smooth manifolds, which contributes to their robustness to domain changes.

Interestingly, recent learning-based methods have also highlighted the importance of both robust discretization-insensitive conversion [18, 8, 35, 13] and of enforcing correct losses during training using either functional [35] or point-to-point correspondences [18, 12, 13]. In the latter category, conversion between functional and point-to-point maps is done as a non-learned layer *in the network* and thus must be correctly and consistently defined. The role of the adjoint was considered very recently in a learning context [22] although that work did not address standard functional map conversion nor draw links to existing methods.

We also note that several works have studied the importance of iterative conversion between pointwise and functional (or, more broadly, probabilistic) correspondences [23, 31, 44, 45]. As the conversion step is performed repeatedly within these approaches, it strongly contributes to the overall final accuracy [23].

Due to the ubiquitous nature of the conversion between functional and point-to-point correspondences, our analysis has direct implications in all of these scenarios. As we demonstrate below, the conversion that we consider and the resulting spectral alignment methods, while based on the same underlying principles as in some previous works [28, 10, 15], is theoretically better justified, more robust and results in practical improvement especially in challenging cases of shapes with different discretizations. Specifically, we remark that earlier approaches for pointwise conversion

like [28, 15] lack a formal discussion of delta functions and the importance of combining them with adjoint operators as we show in Theorem 1. Hence the pointwise conversion schemes in these methods are purely heuristic and ultimately rely on the same recovery approach as [28]. Additionally, we provide a proof of *optimal spectral alignment* which is essential to justify the application of regularised optimal transport in order to enable a fast, accurate and bijective pointwise conversion.

Optimal Transport We also note briefly that optimal transport is another widely-used relaxation for matching problems [40, 21, 45, 9]. Our use of Sinkhorn algorithm is directly inspired by advances in this area [7, 39]. Moreover, as we demonstrate below, existing techniques that use the formalism of optimal transport for solving assignment problems including the Product Manifold Filter with the heat kernels [21, 45] fall within the general spectral alignment formalism that we study. However, in contrast to these existing methods which often rely on large dense matrices like geodesic distances or heat kernels, our formulation of the regularized transport problem is more efficient, as it only involves sparse matrix manimulation. Our use of Sinkhorn's algorithm is thus both robust and provides good results even for non-rigid 3D shapes with non-uniform sampling.

3. Regularized Spectral Alignment

In this section, we provide a formalism for pointwise map conversion from functional maps, and discuss how this is related to the problem of aligning the spectral embedding of shapes. We then propose a general framework for iterative spectral alignment, and illustrate that many previous shape matching refinement methods are regularized variants of our meta algorithm.

3.1. Background and Operators in Smooth settings

Functional maps Suppose we are given two smooth surfaces $\mathcal{X}, \mathcal{Y}$ and a pointwise map $T_{\mathcal{XY}}$ that maps a point in $\mathcal{X}$ to a point in $\mathcal{Y}$. As introduced in [28], the *functional map* associated with the given pointwise map $T_{\mathcal{XY}}$ is defined via pullback (denoted as $T^{\mathcal{F}}_{\mathcal{YX}}$): for any real-valued function $f : \mathcal{Y} \to \mathbb{R}$, its image $g = T^{\mathcal{F}}_{\mathcal{YX}}(f)$ is a real-valued function on $\mathcal{X}$ so that $g(x) = f\big(T_{\mathcal{XY}}(x)\big)$ for any $x \in \mathcal{X}$. Note the change in direction between $T_{\mathcal{XY}}$ and $T^{\mathcal{F}}_{\mathcal{YX}}$.

Although a functional map can be used directly to transport real-valued functions, in most cases, we are also interested in how to recover the pointwise map $T_{\mathcal{XY}}$ from a functional map $T^{\mathcal{F}}_{\mathcal{YX}}$. In the original work [28] (Remark 4.1 in Section 4), the method that is alluded to is to use $T^{\mathcal{F}}_{\mathcal{YX}}$ to map *indicator functions*, i.e. functions that equal 1 at some point and zero elsewhere. Unfortunately, this has a major problem in L^2 (the space of square integrable functions), since such pointwise indicators are equivalent to the zero function. This means that in an orthonormal basis, such as the LB eigenfunctions, such functions will be represented as vectors of zeros. As a result, we cannot apply such a method in practice directly.

A more principled approach can be obtained by using the functional map adjoint as we discuss below.

Adjoint Operator Given a functional map $T^{\mathcal{F}}_{\mathcal{YX}}$, the adjoint functional map operator $T^{\mathcal{A}}_{\mathcal{XY}}$ is a linear operator that maps real-valued functions on $\mathcal{X}$ to those on $\mathcal{Y}$, and is defined implicitly [15], so that $\forall\ f : \mathcal{Y} \to \mathbb{R},\ g : \mathcal{X} \to \mathbb{R}$:

$$\langle\, T^{\mathcal{A}}_{\mathcal{XY}}(g), f \,\rangle_{\mathcal{Y}} = \langle\, g, T^{\mathcal{F}}_{\mathcal{YX}}(f) \,\rangle_{\mathcal{X}} \tag{1}$$

Here we denote with $\langle\ ,\ \rangle_{\mathcal{X}}$ and $\langle\ ,\ \rangle_{\mathcal{Y}}$ the L^2 inner product for functions respectively on shape $\mathcal{X}$ and $\mathcal{Y}$. The adjoint always exists and is unique by the Riesz representation theorem (see also Theorem 3.1 in [15]). Note that the adjoint operator of functional maps has been considered, e.g., in [15] although its role in pointwise map recovery was not explicitly addressed in that work.

Note that the adjoint operator $T^{\mathcal{A}}_{\mathcal{XY}}$, unlike the functional map, maps the functions in the *same* direction as the point wise map $T_{\mathcal{XY}}$. Besides the consistent direction, the adjoint operator has another nice property that it always maps Dirac deltas to Dirac deltas as shown in Theorem 1 below.

Importantly, Dirac deltas are *not* functions but are instead special cases of *distributions*, which are continuous linear functionals over the space of smooth square-integrable functions (also known as test functions). Thus, for a distribution d, given any smooth test function h, $d(h)$ is a real-value. One can construct a distribution d_f from a square-integrable function f via integration: $d_f(h) = \int f(x)h(x)d\mu(x)$. For the special case of the Dirac deltas, i.e., $d = \delta_x$, we have $\delta_x(h) = h(x)$ for any test function h. Then by definition, we write $\langle\, \delta_x, h \,\rangle = h(x)$.

Theorem 1. *Let $T^{\mathcal{A}}_{\mathcal{XY}}$ be the adjoint operator associated with a point-to-point mapping $T_{\mathcal{XY}}$ as in Eq. (1). Then $T^{\mathcal{A}}_{\mathcal{XY}}\delta_x = \delta_{T_{\mathcal{XY}}(x)}$ for all $x \in \mathcal{X}$.*

Proof. Using Eq. (1) for any $f : \mathcal{Y} \to \mathbb{R}$, we get:

$$\langle\, T^{\mathcal{A}}_{\mathcal{XY}}\delta_x, f \,\rangle_{\mathcal{Y}} = \langle\, \delta_x, T^{\mathcal{F}}_{\mathcal{YX}} f \,\rangle_{\mathcal{X}} = \langle\, \delta_x, f \circ T_{\mathcal{XY}} \,\rangle_{\mathcal{X}} = f\big(T_{\mathcal{XY}}(x)\big)$$

Therefore, $T^{\mathcal{A}}_{\mathcal{XY}}\delta_x$ equals some distribution d such that $\langle d, f \rangle_{\mathcal{Y}} = f\big(T_{\mathcal{XY}}(x)\big)$ for any function f on $\mathcal{Y}$. By uniqueness of distributions this means that: $T^{\mathcal{A}}_{\mathcal{XY}}\delta_x = \delta_{T_{\mathcal{XY}}(x)}$. □

To summarize, given a pointwise map $T_{\mathcal{XY}}$, we can obtain a functional map $T^{\mathcal{F}}_{\mathcal{YX}}$. We can then define the adjoint functional map $T^{\mathcal{A}}_{\mathcal{XY}}$. To recover $T_{\mathcal{XY}}$, we first define a Dirac delta function δ_x for each point x on $\mathcal{X}$ and map it using $T^{\mathcal{A}}_{\mathcal{XY}}$. Its image $T^{\mathcal{A}}_{\mathcal{XY}}\delta_x$ gives the Dirac delta function defined at the corresponding point $T_{\mathcal{XY}}(x) \in \mathcal{Y}$ as required.

3.2. Operators and Deltas in Discrete Settings

The discussion above holds in the smooth setting. Our goal now is to demonstrate that the same results hold in the discrete setting and moreover leads to practical algorithms for pointwise map recovery.

We assume that each shape is represented as a triangle mesh. A smooth function f is discretized as a vector $\mathbf{f}$ that is defined at each of the vertices. We also assume to be given a symmetric mass matrix $\mathbf{A}$ so that the functional inner product $\langle\ f, g\ \rangle$ is discretized as $\mathbf{f}^T\mathbf{A}\mathbf{g}$ (see [24]). As is commonly done in geometry processing [37], we will sometimes assume that $\mathbf{A}$ is a diagonal matrix of area-weights. While this assumption not strictly required, it simplifies some of the calculations below.

Any pointwise map $T_{\mathcal{X}\mathcal{Y}}$ from a triangle mesh $\mathcal{X}$ to a triangle mesh $\mathcal{Y}$ can be written as a binary matrix $\Pi_{\mathcal{Y}\mathcal{X}}$ of size $n_\mathcal{X} \times n_\mathcal{Y}$ (number of vertices on shapes $\mathcal{X}, \mathcal{Y}$ respectively) so that $\Pi_{\mathcal{Y}\mathcal{X}}(x, y)$ equals to 1 if $T_{\mathcal{X}\mathcal{Y}}(x) = y$ and equals to 0 otherwise. We can see that, $\Pi_{\mathcal{Y}\mathcal{X}} \in \mathbb{R}^{n_\mathcal{X} \times n_\mathcal{Y}}$ is a discrete functional map in the *full* basis that transfers discrete function $\mathbf{f} \in \mathbb{R}^{n_\mathcal{Y}}$ from $\mathcal{Y}$ to a function $\mathbf{g}$ on $\mathcal{X}$ via matrix multiplication, i.e., $\mathbf{g} = \Pi_{\mathcal{X}\mathcal{Y}}\mathbf{f} \in \mathbb{R}^{n_\mathcal{X}}$.

Discrete Dirac deltas Recall the definition of Dirac deltas in the smooth setting: $< \delta_x, h >= h(x)$ must hold for any test function h. Discretizing this equation on a triangle mesh we get: $\delta_x^T\mathbf{A}\mathbf{h} = \mathbf{h}(x)$ for any function $\mathbf{h}$. Let $\mathbf{e}_x$ denote the indicator at x: i.e., a vector that equals to 1 in the position corresponding to x and zeros elsewhere. We then get $\delta_x^T\mathbf{A}\mathbf{h} = \mathbf{e}_x^T\mathbf{h}$. Since this must hold for all $\mathbf{h}$ and since $\mathbf{A}$ is symmetric, we get: $\mathbf{A}\delta_x = \mathbf{e}_x$, or equivalently $\delta_x = \mathbf{A}^{-1}e_x$. If we assume $\mathbf{A}$ to be diagonal, we obtain that δ_x is a vector such that: $\delta_x(y)$ equals to $1/\mathbf{A}(x)$ if $y = x$ and equals to 0 otherwise. Remark that δ_x is *not* the same as the indicator (also known as the hat) function on the mesh. Instead, we must factor the area of the corresponding point. Intuitively this is because the Delta function is "responsible" for the entire region around a given point.

Discrete Adjoint Operators in Full Basis As defined in the smooth setting, the adjoint operator can be derived from the functional map via $\langle\ T^\mathcal{A}_{\mathcal{X}\mathcal{Y}}(g), f\ \rangle_\mathcal{Y} = \langle\ g, T^\mathcal{F}_{\mathcal{Y}\mathcal{X}}(f)\ \rangle_\mathcal{X}$. Similarly, the discrete adjoint operator in the full basis, $\Gamma_{\mathcal{X}\mathcal{Y}} \in \mathbb{R}^{n_\mathcal{Y} \times n_\mathcal{X}}$ is a matrix that maps discrete functions from $\mathcal{X}$ to $\mathcal{Y}$, i.e., $\langle\ \Gamma_{\mathcal{X}\mathcal{Y}}\mathbf{g}, \mathbf{f}\ \rangle_\mathcal{Y} = \langle\ \mathbf{g}, \Pi_{\mathcal{Y}\mathcal{X}}\mathbf{f}\ \rangle_\mathcal{X}$.

Denoting the area matrices of shapes $\mathcal{X}$, $\mathcal{Y}$ as $\mathbf{A}_\mathcal{X}, \mathbf{A}_\mathcal{Y}$, we have: $\mathbf{f}^T\mathbf{A}_\mathcal{Y}\Gamma_{\mathcal{X}\mathcal{Y}}\mathbf{g} = \mathbf{f}^T\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{A}_\mathcal{X}\mathbf{g}$, which must hold for all pairs of $\mathbf{f}, \mathbf{g}$. Therefore, we have:

$$\Gamma_{\mathcal{X}\mathcal{Y}} = \mathbf{A}_\mathcal{Y}^{-1}\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{A}_\mathcal{X} \tag{2}$$

If $\mathbf{A}_\mathcal{X}, \mathbf{A}_\mathcal{Y}$ are diagonal matrices, this leads to:

$$\Gamma_{\mathcal{X}\mathcal{Y}}(y, x) = \begin{cases} \mathbf{A}_\mathcal{X}(x)/\mathbf{A}_\mathcal{Y}(y) & \text{if } T_{\mathcal{X}\mathcal{Y}}(x) = y \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Remark that Theorem 1 also holds in the discrete setting. Specifically, for any discrete Dirac delta $\delta_x = \mathbf{A}_\mathcal{X}^{-1}\mathbf{e}_x$ on $\mathcal{X}$, we have $\Gamma_{\mathcal{X}\mathcal{Y}}\delta_x = \mathbf{A}_\mathcal{Y}^{-1}\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{A}_\mathcal{X}\mathbf{A}_\mathcal{X}^{-1}\mathbf{e}_x = \mathbf{A}_\mathcal{Y}^{-1}\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{e}_x = \mathbf{A}_\mathcal{Y}^{-1}\mathbf{e}_{T_{\mathcal{X}\mathcal{Y}}(x)} = \delta_{T_{\mathcal{X}\mathcal{Y}}(x)}$ as required.

Discrete Adjoint Operators in a Reduced Basis Given the constructions above, it is easy to translate them to the setting, where functions are represented through their coefficients in some possibly reduced basis, such as the eigenfunctions of the Laplace-Beltrami operator.

Suppose we are given a basis on triangle meshes $\mathcal{X}, \mathcal{Y}$ that we store as columns of matrices $\Phi_\mathcal{X}, \Phi_\mathcal{Y}$ respectively. We assume that these bases are orthonormal with respect to the corresponding mass matrices $\mathbf{A}_\mathcal{X}, \mathbf{A}_\mathcal{Y}$, i.e., $\Phi_\mathcal{X}^T\mathbf{A}_\mathcal{X}\Phi_\mathcal{X} = Id$ and $\Phi_\mathcal{Y}^T\mathbf{A}_\mathcal{Y}\Phi_\mathcal{Y} = Id$. Note that the basis matrice have size: $\Phi_\mathcal{X} \in \mathbb{R}^{n_\mathcal{X} \times k_\mathcal{X}}$ and $\Phi_\mathcal{Y} \in \mathbb{R}^{n_\mathcal{Y} \times k_\mathcal{Y}}$, where $k_\mathcal{X} \leq n_\mathcal{X}, k_\mathcal{Y} \leq n_\mathcal{Y}$. In practice, the number of basis elements k is in range of $[\,50\,,\,200\,]$, while the number of vertices n can be tens of thousands.

In the reduced basis, a functional map simply "translates" the coefficients of functions expressed in the given basis. Specifically, following [29] we have that a functional map in the reduced is simply given as: $\mathbf{C}_{\mathcal{Y}\mathcal{X}} = \Phi^\dagger_\mathcal{X}\Pi_{\mathcal{Y}\mathcal{X}}\Phi_\mathcal{Y}$. Note that $\mathbf{C}_{\mathcal{Y}\mathcal{X}} \in \mathbb{R}^{k_\mathcal{X} \times k_\mathcal{Y}}$ is a functional map in the reduced basis mapping coefficients of functions from $\mathcal{Y}$ to $\mathcal{X}$. Applying the same idea to the adjoint operator, we obtain the adjoint operator in the reduced basis, i.e., $\mathbf{D}_{\mathcal{X}\mathcal{Y}} = \Phi^\dagger_\mathcal{Y}\Gamma_{\mathcal{X}\mathcal{Y}}\Phi_\mathcal{X}$. Plugging in Eq. (2), we have

$$\begin{aligned} \mathbf{D}_{\mathcal{X}\mathcal{Y}} &= \Phi_\mathcal{Y}^{-1}\Gamma_{\mathcal{X}\mathcal{Y}}\Phi_\mathcal{X} = \Phi^\dagger_\mathcal{Y}\mathbf{A}_\mathcal{Y}^{-1}\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{A}_\mathcal{X}\Phi_\mathcal{X} \\ &= \Phi_\mathcal{Y}^T\Pi^T_{\mathcal{Y}\mathcal{X}}\left(\Phi^\dagger_\mathcal{X}\right)^T = \mathbf{C}^T_{\mathcal{Y}\mathcal{X}} \end{aligned} \tag{4}$$

Here we used the fact that $\Phi^\dagger_\mathcal{S} = \Phi^T_\mathcal{S}\mathbf{A}_\mathcal{S}$ ($\mathcal{S} = \mathcal{X}, \mathcal{Y}$) since the basis is assumed to be orthonormal. This calculation shows that $\mathbf{D}$ in the reduced basis is simply the transpose of the functional map $\mathbf{C}$ in the opposite direction.

We note that Theorem 1 also holds approximately in the discrete setting with the reduced basis. Recall that the discrete Dirac delta on shape $\mathcal{X}$ in the full basis is $\delta_x = \mathbf{A}_\mathcal{X}^{-1}\mathbf{e}_x$. We can compute its corresponding coefficient vector $\mathbf{d}_x$ w.r.t. the reduced basis $\Phi_\mathcal{X}$, i.e., $\mathbf{d}_x = \Phi^\dagger_\mathcal{X}\mathbf{A}_\mathcal{X}^{-1}\mathbf{e}_x = \Phi_\mathcal{X}^T\mathbf{A}_\mathcal{X}\mathbf{A}_\mathcal{X}^{-1}\mathbf{e}_x = \Phi_\mathcal{X}^T\mathbf{e}_x$ We then have:

$$\begin{aligned} \mathbf{D}_{\mathcal{X}\mathcal{Y}}\mathbf{d}_x &= \left(\Phi^\dagger_\mathcal{Y}\mathbf{A}_\mathcal{Y}^{-1}\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{A}_\mathcal{X}\Phi_\mathcal{X}\right)\Phi_X^T\mathbf{e}_x \\ &= \Phi_\mathcal{Y}^T\mathbf{A}_\mathcal{Y}\mathbf{A}_\mathcal{Y}^{-1}\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{A}_\mathcal{X}\Phi_\mathcal{X}\Phi_X^T\mathbf{e}_x = \Phi_\mathcal{Y}^T\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{A}_\mathcal{X}\Phi_X\Phi_X^T\mathbf{e}_x \\ &\approx \Phi_\mathcal{Y}^T\Pi^T_{\mathcal{Y}\mathcal{X}}\mathbf{e}_x = \Phi_\mathcal{Y}^T\mathbf{e}_{T_{\mathcal{X}\mathcal{Y}}(x)} = \mathbf{d}_{T_{\mathcal{X}\mathcal{Y}}(x)} \end{aligned}$$

Here, the approximation error comes from the functional basis truncation. Since $\Phi^T\left(\mathbf{A}\Phi\Phi^T - Id\right) = 0$ holds, $\mathbf{e}_x$ can be considered as an approximation for $\mathbf{A}_\mathcal{X}\Phi_\mathcal{X}\Phi_\mathcal{X}^T\mathbf{e}_x$. We therefore have $\mathbf{D}_{\mathcal{X}\mathcal{Y}}\mathbf{d}_x \approx \mathbf{d}_{T_{\mathcal{X}\mathcal{Y}}(x)}$.

In summary, in this section, we justified that the adjoint operators in the discrete setting, both in the full basis or

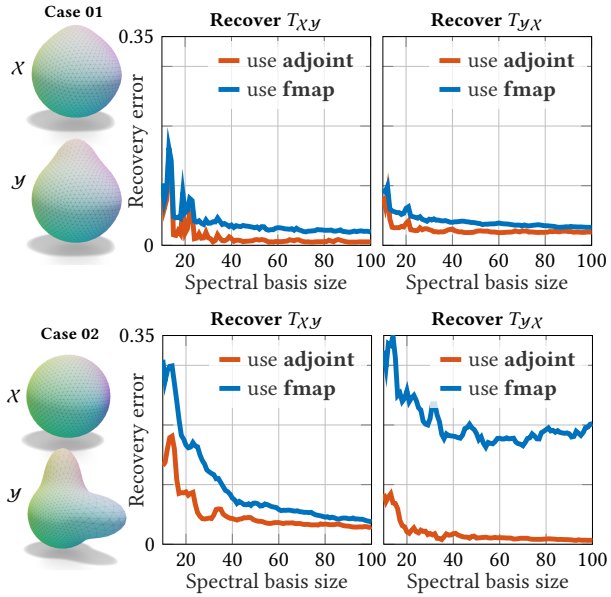

Figure 2: *Top*: a pair of near-isometric deformed spheres. *Bottom*: a pair of non-isometric deformed spheres. We compare the *pointwise map recover error* between using the adjoint operators (Eq. (5) and (6)) and using the functional maps as suggested in [28].

reduced basis, lead to the same result as highlighted in Theorem 1. Specifically, the discrete adjoint operators always map the discrete Dirac deltas (coefficients) to Dirac deltas (coefficients). Note that the same claim *does not hold* for the functional map itself, both because the map direction is reversed and because area elements might be different.

3.3. Pointwise Map Conversion

Overview Recall that in the smooth setting, we can always convert a given pointwise map into a functional map and then use Eq. (1) to obtain the adjoint operator; we can then recover the original pointwise map from an adjoint operator as shown in Theorem 1 without any additional assumptions on the map (e.g., isometric or local volume preserving maps). This one-to-one relationship between pointwise maps and the adjoint functional map also holds exactly in the discrete setting in both the full and reduced basis, up to basis truncation errors in the latter.

This suggests the following pipeline for computing pointwise correspondences: first we can use existing methods [28, 31, 30] to compute a functional map $\mathbf{C}_{\mathcal{YX}}$ through optimization using some descriptors. Then the adjoint operator can be obtained by setting $\mathbf{D}_{\mathcal{XY}} = \mathbf{C}_{\mathcal{YX}}^T$. Finally, we can extract a pointwise map $T_{\mathcal{XY}}$ from $\mathbf{D}_{\mathcal{XY}}$ by using $\mathbf{D}_{\mathcal{XY}}$ to map the coefficients of the Dirac deltas.

Map conversion We now discuss in detail how to extract a pointwise map $T_{\mathcal{XY}}$ from an adjoint functional map $\mathbf{D}_{\mathcal{XY}}$. Recall that we have $\mathbf{D}_{\mathcal{XY}}\mathbf{d}_x \approx \mathbf{d}_{T_{\mathcal{XY}}(x)}$, which can

Algorithm 1: Iterative Meta Algorithm (IMA)

Input : A pair of shapes $\mathcal{X}, \mathcal{Y}$ with basis $\Phi_{\mathcal{X}}, \Phi_{\mathcal{Y}}$
Output: Pointwise map $T_{\mathcal{XY}}$ and adjoint map $\mathbf{D}_{\mathcal{XY}}$
Initialization : An initial guess of $\mathbf{D}_{\mathcal{XY}}$
while *Not converged* **do**
  (1) Extract map $T_{\mathcal{XY}}$ from $\mathbf{D}_{\mathcal{XY}}$ (e.g. Eq. (5))
  (2) Convert the extracted $T_{\mathcal{XY}}$ to $\mathbf{D}_{\mathcal{XY}}$ (Eq. (4))
end

be equivalently written as $\mathbf{D}_{\mathcal{XY}}\Phi_{\mathcal{X}}^T\mathbf{e}_x \approx \Phi_{\mathcal{Y}}^T\mathbf{e}_{T_{\mathcal{XY}}(x)}$. Therefore, we have $T_{\mathcal{XY}}(x) = \text{NNsearch}\big(\Phi_{\mathcal{Y}}, \mathbf{e}_x^T\Phi_{\mathcal{X}}\mathbf{D}_{\mathcal{XY}}^T\big)$, where $\text{NNsearch}\big(\mathbf{P}, \mathbf{q}\big)$ returns the closest point (nearest neighbor) in $\mathbf{P}$ for the query point $\mathbf{q}$, where points in $\mathbf{P}$ are stored in rows. We then iterate through all the points x in shape $\mathcal{X}$ and obtain the pointwise map:

$$T_{\mathcal{XY}} = \text{NNsearch}\big(\Phi_{\mathcal{Y}}, \Phi_{\mathcal{X}}\mathbf{D}_{\mathcal{XY}}^T\big) \tag{5}$$

We can similarly extract a pointwise map $T_{\mathcal{YX}}$ in the opposite direction from $\mathbf{D}_{\mathcal{XY}}$ via:

$$T_{\mathcal{YX}} = \text{NNsearch}\big(\Phi_{\mathcal{X}}\mathbf{D}_{\mathcal{XY}}^T, \Phi_{\mathcal{Y}}\big) \tag{6}$$

We emphasize that this is different from the pointwise map conversion proposed in [28] and used in follow-up works including [15], where the functional map matrix $\mathbf{C}_{\mathcal{YX}}$ is used directly to transport Delta functions. As remarked above, this procedure, unfortunately has no justification in the smooth setting.

Synthetic example Fig. 2 shows two pairs of deformed spheres *with different underlying mesh structure*, and we compare the map conversion error between using the adjoint operators and the functional maps when transferring Delta functions. Remark that for a pair of shapes that are near-isometric (top of Fig. 2), implying that local areas are preserved by the underlying map, using functional maps to recover the pointwise correspondence is comparable (while nevertheless being slightly worse) to the results of using the adjoint operators. However, when two shapes are far from isometry (bottom of Fig. 2), using the adjoint operators achieves significantly better results than using the functional maps, which may not even converge with increasing basis size (bottom right).

3.4. Spectral Embedding Alignment

The discussion above focuses on the functional map representation and the conversion step between functional maps and point-to-point ones. Another, very fruitful, perspective on the same procedure can be obtained by considering the *spectral embeddings* of the two shapes and alignments between them.

Given a shape $\mathcal{X}$ with the reduced basis $\Phi_{\mathcal{X}}$ we call its spectral embedding the point cloud obtained by constructing a point for each *row* of $\Phi_{\mathcal{X}}$, using each columns as coordinates. From Section 3.1, this is the same as constructing

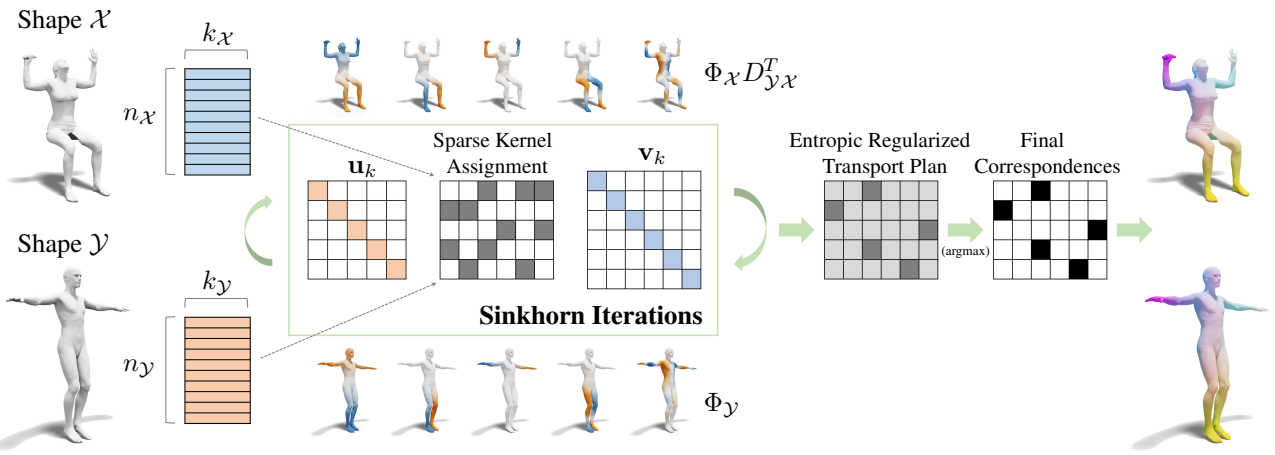

Figure 3: **Fast Sinkhorn Filters:** Illustrating spectral embedding alignment using regularized optimal transport. Our pipeline inputs a set of aligned basis functions and outputs a pointwise correspondence between the shapes.

a point cloud where the coordinates of each point represent the coefficients of its delta function in the reduced basis.

Optimal linear alignment Recall from the discussion above we have $\mathbf{D}_{\mathcal{XY}}\mathbf{d}_x \approx \mathbf{d}_{T_{\mathcal{XY}}(x)}$. Since $\mathbf{d}_x$ represents the coefficients of δ_x in the reduced basis of shape $\mathcal{X}$, and $\mathbf{D}_{\mathcal{XY}}\mathbf{d}_x$ represents the coefficients of the image in the reduced basis, this means that the operator $\mathbf{D}_{\mathcal{XY}}$ *aligns the spectral embeddings of the shape* $\mathcal{X}$ *and* $\mathcal{Y}$. This can also be written as follows: $\mathbf{D}_{\mathcal{XY}}\Phi_{\mathcal{X}}^T \approx (\Pi_{\mathcal{YX}}\Phi_{\mathcal{Y}})^T$.

Interestingly, the adjoint operator can also be deduced as *the optimal linear transformation that aligns the spectral embeddings*, given a point-to-point map. I.e.,

$$\mathbf{D}_{\mathcal{XY}} = \arg\min_X \left\| X\Phi_{\mathcal{X}}^T - \left(\Pi_{\mathcal{YX}}\Phi_{\mathcal{Y}}\right)^T \right\| \tag{7}$$

which can be easily demonstrated via:

$$\begin{aligned} X^* &= \arg\min_X \left\| X\Phi_{\mathcal{X}}^T - \left(\Pi_{\mathcal{YX}}\Phi_{\mathcal{Y}}\right)^T \right\| \\ &= \arg\min_X \left\| \Phi_{\mathcal{X}} X^T - \Pi_{\mathcal{YX}}\Phi_{\mathcal{Y}} \right\| \\ &= \left(\Phi_X^\dagger \Pi_{\mathcal{YX}}\Phi_{\mathcal{Y}}\right)^T = \mathbf{C}_{\mathcal{YX}}^T = \mathbf{D}_{\mathcal{XY}} \end{aligned} \tag{8}$$

Intuitively this means that the adjoint is the *optimal* linear operator that aligns two spectral embeddings given a pointwise map. Again, we stress that the same interpretation does not hold for the functional map itself, due to the reasons mentioned above.

Iterative meta algorithm (IMA) As remarked above the spectral embedding $\Phi_{\mathcal{X}}$ of a shape $\mathcal{X}$ can be interpreted as a point cloud in $k_{\mathcal{X}}$ dimensional space. Further, the adjoint functional map $\mathbf{D}_{\mathcal{XY}}$ is the optimal linear transformation that aligns the spectral embeddings of $\mathcal{X}$ and $\mathcal{Y}$ given a point-to-point map.

Since *a priori* we do not have access to either the point-to-point map or the functional map (or its adjoint), this suggests the following iterative procedure: we first estimate an initial functional (or adjoint) map; we then iteratively extract a pointwise map $T_{\mathcal{XY}}$ from the adjoint map $\mathbf{D}_{\mathcal{XY}}$, and, again, compute the adjoint map $\mathbf{D}_{\mathcal{XY}}$ from the obtained pointwise map $T_{\mathcal{XY}}$. We call this scheme an Iterative Meta Algorithm (summarized in Algorithm 1).

Regularized spectral alignment In practice, this simple procedure does not work well though the convergence is guaranteed for the following reasons: (1) The dimensionality of the spectral embedding is typically quite high. This means that there could be many local minima in the alignment energy. (2) This procedure provides no guarantees on *what kind* of map (pointwise or functional) recovered. For example in the full basis (i.e., when the number of basis functions equals the number of points) there is a linear transformation for *any* point-to-point map that aligns the spectral embeddings perfectly. Thus, this procedure will terminate in a single iteration. For these reasons, some additional information must be injected into this basic method. For example, some existing methods such as ICP [28], PMF [45], ZoomOut [23] can be cast as variants of IMA with additional regularization.

Specifically, **ICP** [28] promotes *orthonormal* functional maps by projecting the singular values of the functional map to 1 at each iteration, which correspond to locally *area-preserving* correspondences. **PMF** [45] introduces *bijectivity* as a hard constraint for pointwise map conversion which turns the procedure to an assignment problem and is solved using the auction algorithm. At the same time, effectively by varying the time parameter of the kernel matrices involved, the size of basis used for spectral embedding is progressively increased between iterations. **ZoomOut** [23] promotes *orthonormal* functional maps *for each* principal submatrix via increasing the size of basis for the spectral embedding progressively between iterations. See appendix for pseudo-code of the mentioned algorithms.

Algorithm 2: Fast Sinkhorn Filtering

Input : $\Phi_{\mathcal{X}}, \Phi_{\mathcal{Y}}, \mathbf{D}_{\mathcal{XY}}$, Parameters λ, k_0, N_0
Output: Pointwise maps $\Pi_{\mathcal{XY}}$ and $\Pi_{\mathcal{YX}}$
Step1: Populate Sparse Kernel Assignment Matrix $\mathbf{K}^{\lambda} \in \mathbb{R}^{n_{\mathcal{X}} \times n_{\mathcal{Y}}}$ as follows:

1. Align the spectral basis functions: $\mathcal{F}_{\mathcal{X}} = \Phi_{\mathcal{X}} \mathbf{D}_{\mathcal{YX}}^{T} \in \mathbb{R}^{n_{\mathcal{X}} \times k}$ and $\mathcal{F}_{\mathcal{Y}} = \Phi_{\mathcal{Y}} \in \mathbb{R}^{n_{\mathcal{Y}} \times k}$
2. For each row in $\mathcal{F}_{\mathcal{X}}$, find the k_0 nearest neighbors in $\mathcal{F}_{\mathcal{Y}}$ and let: $d_{ij} = \left\| \mathcal{F}_{\mathcal{X}}(i,:) - \mathcal{F}_{\mathcal{Y}}(j,:) \right\|$ be the aligned spectral distance between the i$^{\text{th}}$ point in shape $\mathcal{X}$ to the j$^{\text{th}}$ point in shape $\mathcal{Y}$. Set $\mathbf{K}_{ij}^{\lambda} = e^{-\lambda\, d_{ij}^2}$

Step2: Estimate Regularized Transport Plan $\mathbf{P}_{\mathcal{XY}}$
Set $a = \frac{1}{n_{\mathcal{Y}}} \mathbb{1}_{n_{\mathcal{Y}}}, b = \frac{1}{n_{\mathcal{X}}} \mathbb{1}_{n_{\mathcal{X}}}$, Initialize:$u_0 = a$
for $k = 1, 2, \ldots N_0$ **do**
  $v_k \leftarrow a / \left(\mathbf{K}^{\lambda^{T}} u_k \right)$ #elementwise division
  $u_k \leftarrow b / (\mathbf{K}^{\lambda} v_k)$
end
$\mathbf{P}_{\mathcal{XY}} = \text{Diag}(u_{N_0})\ \mathbf{K}^{\lambda}\ \text{Diag}(v_{N_0})$
Step3: Extract vertex-to-vertex maps
$\Pi_{\mathcal{XY}} = \text{argmax}(\mathbf{P}_{\mathcal{XY}}^{T}), \Pi_{\mathcal{YX}} = \text{argmax}(\mathbf{P}_{\mathcal{XY}})$

4. Fast Sinkhorn Filters

We propose to solve the spectral embedding alignment as a linear assignment problem such that the pointwise map Π is a *doubly stochastic* matrix with search space $\mathcal{Q}$. Therefore, we can formulate our problem as:

$$\mathbf{P}_{\mathcal{XY}} = \arg\min_{\Pi \in \mathcal{Q}} \left\langle d, \Pi \right\rangle_F \tag{9}$$

where $\mathcal{Q} = \left\{ \Pi \,\middle|\, \Pi \in \mathbb{R}_{\geq 0}^{n_{\mathcal{X}} \times n_{\mathcal{Y}}}, \Pi \mathbb{1}_{n_{\mathcal{Y}}} = \mu_{\mathcal{X}}, \Pi^{T} \mathbb{1}_{n_{\mathcal{X}}} = \mu_{\mathcal{Y}} \right\}$; $\mu_{\mathcal{X}}$ and $\mu_{\mathcal{Y}}$ are initial masses for $\mathcal{X}$ and $\mathcal{Y}$ predefined on each vertex, and $d \in \mathbb{R}^{n_X \times n_Y}$ is a matrix of the pairwise euclidean distances between the aligned embeddings:

$$d_{ij} = \left\| \Phi_{\mathcal{X}}(i,:) D_{\mathcal{XY}}^{T} - \Phi_{\mathcal{Y}}(j,:) \right\| \tag{10}$$

To solve the above problem efficiently, we use the well-known Sinkhorn algorithm that comes from optimal transport theory, which is a tool that allows the computation of distances between functions in a common domain. Specifically, given two probability distributions in a common metric space, the optimal transport distance is the cumulative effort required to shift the mass from each location of the first distribution to some location in the second distribution such that the linear assignment cost is minimized. Importantly, an adjustment to the linear assignment cost of Eq. (11) allows for a much faster and computationally superior numerical solver called the Sinkhorn Algorithm. Therefore, we want to solve:

$$\mathbf{P}_{\mathcal{XY}} = \arg\min_{\Pi \in \mathcal{Q}} \left\langle d, \Pi \right\rangle_F - \lambda\, H(\Pi) \tag{11}$$

Table 1: Comparing different pointwise map conversion methods. We measure different metrics on the recovered maps from 200 pairs of FAUST regular (left) and 190 pairs of FAUST remeshed (right).

Methods	Accuracy ($\times 10^{-3}$)	Bijective ($\times 10^{-3}$)	Coverage (%)	Smoothness	Runtime (s)
Auction	**3.6** / 55	**0** / **0**	**100** / **99**	5.7 / 52.7	11.7 / 5.9
NN	9.5 / 42	8.2 / 27	75.8 / 47.8	4.8 / **6.5**	**0.5** / **0.2**
CPD	7.6 / **32**	10 / 25	82.6 / 71.1	**4.7** / 6.8	13.3 / 6.5
Sinkhorn	4.9 / 33	1.7 / 7	93.6 / 79.7	5.5 / 11.1	2.1 / 1.4

where $H(\Pi) = -\sum_{i,j} \pi_{ij} \log(\pi_{ij})$. We can see that when $\lambda = 0$, this problem is equivalent to the original spectral alignment problem Eq. (9). As discussed in [39], with nonzero λ, $-H(\Pi)$ makes the energy Eq. (11) strictly *convex*, and therefore a unique minimizer exists.

The input to a regularized transportation problem is a $n_{\mathcal{X}}$-by-$n_{\mathcal{Y}}$ distance matrix. We first compute a kernel assignment matrix from the input pairwise distances:

$$\mathbf{K}^{\lambda} = \left[k_{ij}^{\lambda} = e^{-\frac{1}{\lambda} d_{ij}^2} \right]_{i=1,2..n_{\mathcal{X}},\ j=1,2..n_{\mathcal{Y}}} \tag{12}$$

where the distances d_{ij} are defined in Eq. (10). This matrix is then iteratively subject to a matrix scaling procedure leading to a regularized transport plan. See Figure 3.

Note that computing and storing a distance matrix of dimension $n_{\mathcal{X}} \times n_{\mathcal{Y}}$ can be a very time and memory consuming task, especially when the resolution of the shapes is very large. In addition, the value of the kernel in Eq. (12) is significant only for distances that are quite small. Taking advantage of this simple insight, we modify the kernel function of Eq. (12) and construct a *sparse kernel assignment matrix* that is populated by the distance values for only a few nearest neighbors for each point:$\mathbf{K}^{\lambda} = \left[k_{ij}^{\lambda} \right]_{i=1,2..n_{\mathcal{X}},\ j \in \mathcal{N}\{i\}}$.

See Algorithm 2 for a detailed outline. Our construction of the sparse kernel in Sinkhorn iterations leads to improved accuracy and bijectivity without incurring a large runtime. Therefore our solution is a competitive combination of fast, accurate and bijective simultaneously which is in contrast to prior pointwise registration methods as shown in Table 1.

5. Experiments and Evaluation

We evaluate the different pointwise conversion algorithms: Nearest Neighbor, Auction [2], Coherent Point Drift [25], and our Sinkhorn Filter on 200 pairs of FAUST [5] and 190 test pairs of the FAUST remeshed datasets [31]. The auction algorithm [2] solves for a *permutation matrix* for the linear assignment problem. The coherent point drift algorithm is a point set registration technique that models the correspondence as a probability density which is optimized via expectation maximization [25, 33, 34]. All 4 registra-

Table 2: Here we show a quantitative evaluation on 300 FAUST regular shape pairs (left) v.s. 190 FAUST remeshed pairs and 153 SCAPE remeshed pairs (right). We compare our methods (ICP with Sinkhorn and ZoomOut with Sinkhorn) with the baselines across different metrics.

Measurements / Methods	Geometric Metrics				Functional Metrics				Average Runtime (s)
	Accuracy ($\times 10^{-3}$)	Bijectivity ($\times 10^{-3}$)	Coverage	Smoothness	Orthogonality	Laplacian Commutativity	ZoomOut Energy	Chamfer Distance	
Ini	67.3 / 46.5	80.1 / 30.2	41.3 / 49.5	9.54 / 6.88	11.9 / 2.49	353 / 767	11.8 / 6.58	5.29 / 3.85	- / -
ICP	76.0 / 30.4	75.0 / 10.2	76.8 / 76.4	8.09 / 5.11	1.38 / 1.07	224 / 493	4.21 / 3.46	2.89 / 2.56	10.2 / **5.32**
ICP (sink)	68.6 / 29.5	4.38 / 8.07	90.1 / 87.3	13.0 / 5.89	1.42 / 1.21	255 / 448	4.69 / 3.38	2.92 / 2.42	30.4 / 15.8
Deblur	61.9 / 44.4	75.0 / 22.4	39.9 / 43.7	7.62 / 4.80	11.2 / 2.63	361 / 805	12.1 / 6.79	4.64 / 3.31	10.9 / 10.4
RHM	41.9 / 32.0	22.7 / 15.1	78.3 / 76.5	4.28 / **3.40**	4.00 / 1.91	273 / 647	6.01 / 4.51	3.91 / 2.89	41.4 / 47.4
PMF	26.4 / 86.4	**1.99** / 37.7	**100 / 100**	24.0 / 34.9	**1.11** / 2.44	164 / 576	4.02 / 7.97	2.72 / 4.09	737 / 312
BCICP	21.6 / 26.4	4.48 / 12.6	88.9 / 77.6	4.73 / 4.22	1.21 / 1.00	186 / **404**	3.38 / 3.24	2.52 / 2.42	184 / 364
ZoomOut	**15.8 / 22.7**	13.6 / **6.47**	88.0 / 82.1	**3.49** / 3.46	1.32 / **0.99**	**153** / 405	**3.18 / 2.95**	**1.89 / 2.16**	**9.60** / 6.49
ZoomOut (sink)	**12.6 / 20.8**	**1.57 / 6.44**	93.9 / 88.5	**3.44 / 3.40**	1.37 / **0.99**	**148 / 394**	3.21 / 3.07	1.91 / 2.17	28.2 / 19.08

tion methods were initialized with the same pair of basis that were aligned with the adjoint map in Eq. (5).

Table 1 highlights the performance of each of these algorithms that are essential components to almost all correspondence methods including prominent recent ones like [9, 23, 45]. We measure the following metrics: accuracy, bijectivity, smoothness, coverage and average runtime for each conversion. Please refer to the supplementary for detailed definitions of these metrics. Notice that the auction algorithm although superior in a very ideal setting of equal sampling and identical connectivity, is not robust to a change in the discretization of the surface. On the other hand, even though the nearest neighbor has the smallest runtime, it suffers from poor accuracy, bijectivity and coverage. The coherent point drift also suffers from a similar drawback of poor bijectivity. In contrast, the Fast Sinkhorn Filter shows an accurate and bijective output within a very reasonable runtime and these properties are robust to the remeshing of the underlying surface.

We further demonstrate the efficacy of our Sinkhorn filter by using it in conjunction with two prominent variants of the iterative meta algorithm (Algorithm 1): ICP [28] and ZoomOut[23]. We replace the nearest neighbor algorithm in both methods with the Sinkhorn filter and compare the Sinkhornized versions of ICP and Zoomout with previous competing refinement methods on the FAUST regular [5] and FAUST and SCAPE remeshed [31] datasets as shown in Table 2 and visualized in Figure 4. We compare extensively using the commonly used geometric and functional metrics. Please see the detailed definitions in the appendix. The sinkhornized versions of ICP and ZoomOut are both attributed with: better accuracy (20% improvement on FAUST regular and 8% on remeshed datasets), and better bijectivity in the maps w.r.t. the original ICP/ZoomOut.

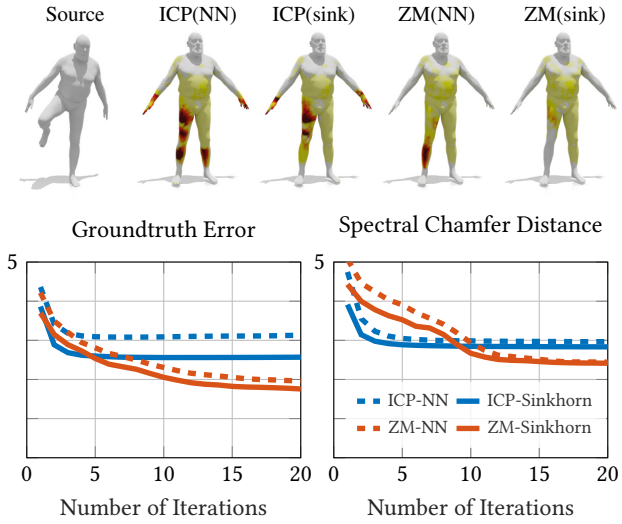

Figure 4: **Sinkhornizing ICP and Zoomout:** Comparison between the original and *sinkhornized* ICP and Zoomout refinement algorithms. (top) Pointwise map errors visualized on target shape. Replacing the nearest neighbor conversion step inside ICP/Zoomout with our Fast Sinkhorn Filter achieves improved accuracy and better spectral alignment.

6. Conclusion

In conclusion, in this paper we propose a theoretical foundation to the problem of pointwise conversion of functional maps and discuss its connection to regularized spectral alignment. Based on this foundation we propose a novel spectral registration technique using optimal transport for spectral alignment, and demonstrate that it improves accuracy and bijectivity of correspondences both independently as well as in conjunction with existing iterative meta algorithms. One limitation of the Sinkhorn filter is the poor generalization to the challenging cases of partiality which we would like to address in future work.

Acknowledgements Parts of this work were supported by the KAUST OSR Award No. CRG-2017-3426, the ERC Starting Grant No. 758800 (EXPROTEA), the ANR AI Chair AIGRETTE and the ERC Starting Grant No. 802554 (SPECGEO).

References

[1] Yonathan Aflalo and Ron Kimmel. Spectral multidimensional scaling. *PNAS*, 110(45):18052–18057, 2013. 2

[2] Dimitri P Bertsekas. The auction algorithm: A distributed relaxation method for the assignment problem. *Annals of operations research*, 14(1):105–123, 1988. 7

[3] Silvia Biasotti, Andrea Cerri, Alex Bronstein, and Michael Bronstein. Recent trends, applications, and perspectives in 3d shape similarity assessment. *Computer Graphics Forum*, 35(6):87–119, 2016. 2

[4] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. FAUST: Dataset and evaluation for 3D mesh registration. In *Proc. CVPR*, pages 3794–3801, Columbus, Ohio, 2014. IEEE. 1

[5] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014. 7, 8

[6] Oliver Burghard, Alexander Dieckmann, and Reinhard Klein. Embedding shapes with Green's functions for global shape matching. *Computers & Graphics*, 68:1–10, 2017. 2

[7] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013. 3

[8] Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8592–8601, 2020. 1, 2

[9] Marvin Eisenberger, Aysim Toker, Laura Leal-Taixé, and Daniel Cremers. Deep shells: Unsupervised shape correspondence with optimal transport, 2020. 3, 8

[10] Danielle Ezuz and Mirela Ben-Chen. Deblurring and denoising of maps between shapes. *Computer Graphics Forum*, 36(5):165–174, 2017. 1, 2

[11] Danielle Ezuz and Mirela Ben-Chen. Deblurring and denoising of maps between shapes. *Computer Graphics Forum*, 36(5):165–174, 2017. 1, 2

[12] Dvir Ginzburg and Dan Raviv. Cyclic functional mapping: Self-supervised correspondence between non-isometric deformable shapes. In *European Conference on Computer Vision*, pages 36–52. Springer, 2020. 2

[13] Oshri Halimi, Or Litany, Emanuele Rodola, Alex M Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4370–4379, 2019. 1, 2

[14] Qixing Huang, Fan Wang, and Leonidas Guibas. Functional map networks for analyzing and exploring large shape collections. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014. 1

[15] Ruqi Huang and Maks Ovsjanikov. Adjoint map representation for shape analysis and matching. *Computer Graphics Forum*, 36(5):151–163, 2017. 1, 2, 3, 5

[16] Artiom Kovnatsky, Michael Bronstein, Alex Bronstein, Klaus Glashoff, and Ron Kimmel. Coupled quasi-harmonic bases. *Computer Graphics Forum*, 32(2pt4):439–448, 2013. 1, 2

[17] Artiom Kovnatsky, Michael M Bronstein, Xavier Bresson, and Pierre Vandergheynst. Functional correspondence by matrix completion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 905–914, 2015. 2

[18] Or Litany, Tal Remez, Emanuele Rodolà, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5659–5667. IEEE, 2017. 1, 2

[19] Or Litany, Emanuele Rodolà, Alex Bronstein, and Michael Bronstein. Fully spectral partial shape matching. *Computer Graphics Forum*, 36(2):247–258, 2017. 2

[20] Or Litany, Emanuele Rodolà, Alexander M Bronstein, Michael M Bronstein, and Daniel Cremers. Non-rigid puzzles. In *Computer Graphics Forum*, volume 35, pages 135–143. Wiley Online Library, 2016. 2

[21] Manish Mandad, David Cohen-Steiner, Leif Kobbelt, Pierre Alliez, and Mathieu Desbrun. Variance-minimizing transport plans for inter-surface mapping. *ACM Transactions on Graphics*, 36:14, 2017. 3

[22] Riccardo Marin, Marie-Julie Rakotosaona, Simone Melzi, and Maks Ovsjanikov. Correspondence learning via linearly-invariant embedding, 2020. 2

[23] Simone Melzi, Jing Ren, Emanuele Rodolà, Abhishek Sharma, Peter Wonka, and Maks Ovsjanikov. Zoomout: Spectral upsampling for efficient shape correspondence. *ACM Transactions on Graphics (TOG)*, 38(6):155:1–155:14, Nov. 2019. 1, 2, 6, 8

[24] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and mathematics III*, pages 35–57. Springer, 2003. 2, 4

[25] Andriy Myronenko and Xubo Song. Point set registration: Coherent point drift. *IEEE transactions on pattern analysis and machine intelligence*, 32(12):2262–2275, 2010. 7

[26] Dorian Nogneng, Simone Melzi, Emanuele Rodolà, Umberto Castellani, M Bronstein, and Maks Ovsjanikov. Improved functional mappings via product preservation. In *Computer Graphics Forum*, volume 37, pages 179–190. Wiley Online Library, 2018. 2

[27] Dorian Nogneng and Maks Ovsjanikov. Informative descriptor preservation via commutativity for shape matching. *Computer Graphics Forum*, 36(2):259–267, 2017. 2

[28] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)*, 31(4):30:1–30:11, 2012. 1, 2, 3, 5, 6, 8

[29] Maks Ovsjanikov, Etienne Corman, Michael Bronstein, Emanuele Rodolà, Mirela Ben-Chen, Leonidas Guibas, Frederic Chazal, and Alex Bronstein. Computing and processing correspondences with functional maps. In *ACM SIGGRAPH 2017 Courses*, pages 5:1–5:62, 2017. 2, 4

[30] Jing Ren, Mikhail Panine, Peter Wonka, and Maks Ovsjanikov. Structured regularization of functional map com-

putations. In *Computer Graphics Forum*, volume 38, pages 39–53. Wiley Online Library, 2019. 5

[31] Jing Ren, Adrien Poulenard, Peter Wonka, and Maks Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *ACM Transactions on Graphics (TOG)*, 37(6), 2018. 2, 5, 7, 8

[32] Emanuele Rodolà, Luca Cosmo, Michael M Bronstein, Andrea Torsello, and Daniel Cremers. Partial functional correspondence. In *Computer Graphics Forum*, volume 36, pages 222–236. Wiley Online Library, 2017. 1, 2

[33] Emanuele Rodolà, Michael Moeller, and Daniel Cremers. Point-wise map recovery and refinement from functional correspondence. In *Proc. Vision, Modeling and Visualization (VMV)*, 2015. 1, 2, 7

[34] Emanuele Rodolà, Michael Möller, and Daniel Cremers. Regularized pointwise map recovery from functional correspondence. In *Computer Graphics Forum*, volume 36, pages 700–711. Wiley Online Library, 2017. 1, 2, 7

[35] Jean-Michel Roufosse, Abhishek Sharma, and Maks Ovsjanikov. Unsupervised deep learning for structured shape matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1617–1627, 2019. 1, 2

[36] R. Rustamov, M. Ovsjanikov, O. Azencot, M. Ben-Chen, F. Chazal, and L. Guibas. Map-based exploration of intrinsic shape differences and variability. *ACM Transactions on Graphics (TOG)*, 32(4):72:1–72:12, July 2013. 2

[37] Raif M Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. In *Proc. SGP*, pages 225–233. Eurographics Association, 2007. 4

[38] Yusuf Sahillioğlu. Recent advances in shape correspondence. *The Visual Computer*, 36(8):1705–1721, 2020. 1, 2

[39] Justin Solomon, Fernando De Goes, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics (TOG)*, 34(4):1–11, 2015. 3, 7

[40] Justin Solomon, Gabriel Peyré, Vladimir G Kim, and Suvrit Sra. Entropic metric alignment for correspondence problems. *ACM Transactions on Graphics (TOG)*, 35(4):72, 2016. 3

[41] Robert W Sumner and Jovan Popović. Deformation transfer for triangle meshes. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 399–405. ACM, 2004. 1

[42] Gary KL Tam, Zhi-Quan Cheng, Yu-Kun Lai, Frank C Langbein, Yonghuai Liu, David Marshall, Ralph R Martin, Xian-Fang Sun, and Paul L Rosin. Registration of 3D point clouds and meshes: a survey from rigid to nonrigid. *IEEE TVCG*, 19(7):1199–1217, 2013. 2

[43] Oliver Van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011. 1, 2

[44] Matthias Vestner, Zorah Lähner, Amit Boyarski, Or Litany, Ron Slossberg, Tal Remez, Emanuele Rodolà, Alex Bronstein, Michael Bronstein, and Ron Kimmel. Efficient deformable shape correspondence via kernel matching. In *3D Vision (3DV), 2017 International Conference on*, pages 517–526. IEEE, 2017. 2

[45] Matthias Vestner, Roee Litman, Emanuele Rodolà, Alex Bronstein, and Daniel Cremers. Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *Proc. CVPR*, pages 6681–6690, 2017. 1, 2, 3, 6, 8

[46] Fan Wang, Qixing Huang, and Leonidas J Guibas. Image cosegmentation via consistent functional maps. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 849–856, 2013. 2

[47] Y Wang, B Liu, K Zhou, and Y Tong. Vector field map representation for near conformal surface correspondence. *Computer Graphics Forum*, 37(6):72–83, 2018. 2